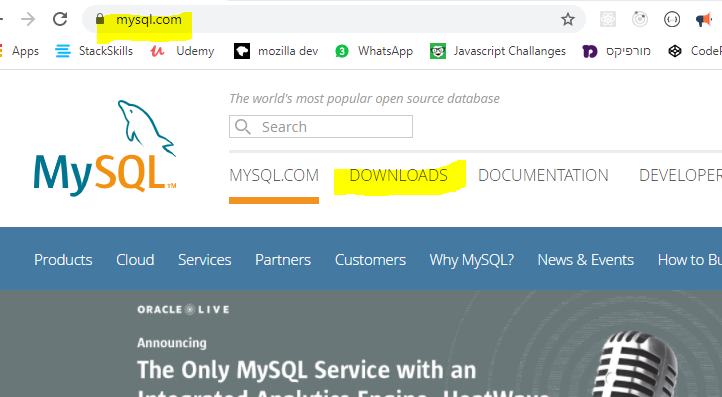
**Derma-Detect home assignment**

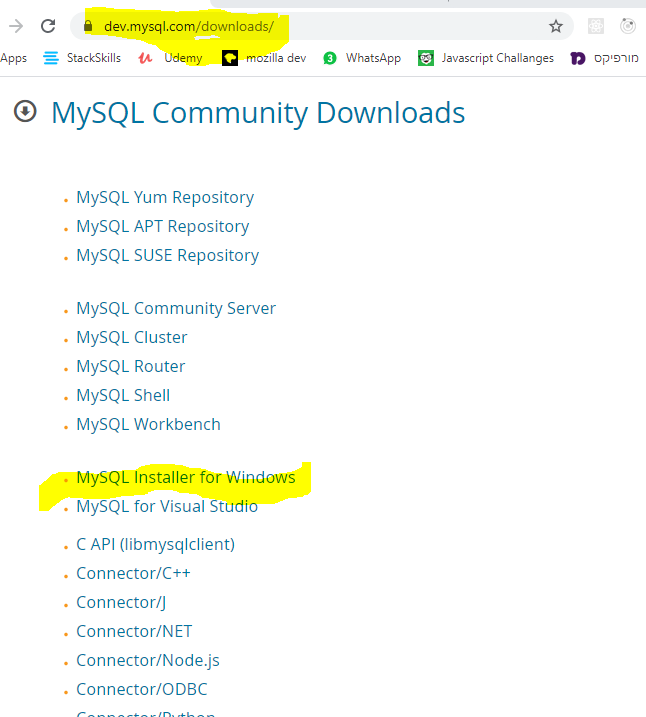
**Let’s create a small CRUD service with MySQL and Token based authentication**

**Make sure you download and install MySQL on your system:**

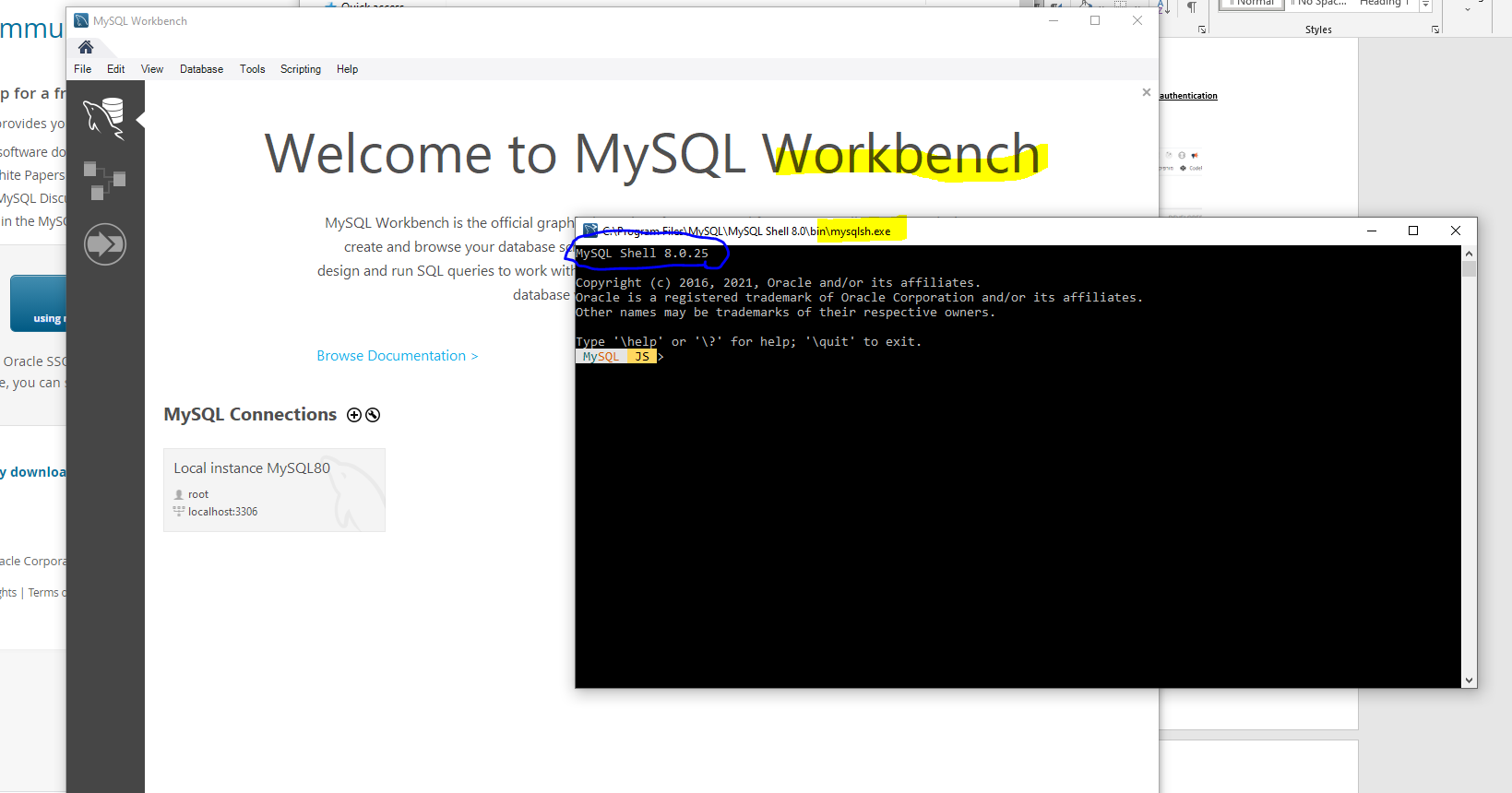
**Setup instructions- https://www.youtube.com/watch?v=OM4aZJW\_Ojs**



Look for the community installer link, and then the windows link:

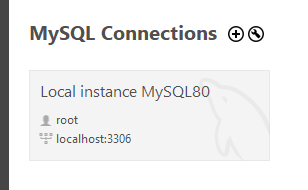


Once you finished the setup the MySQL workbench and shell will be opend:

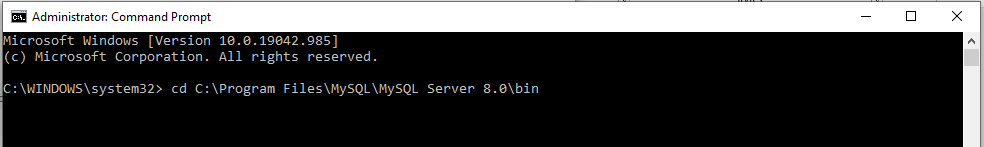


Log in into the root account with the password you set during the setup…

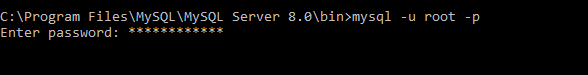
Click it and login:



Open CMD as an admin (by right clicking it, and enter the path to the my sql server bin folder:

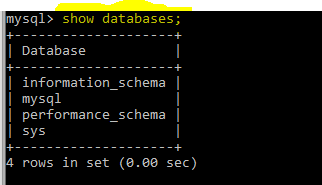


Then : write- mysql -u root -p and enter your DB root user password:



Write >> show databases;

And you will see the defaul DB



**Setup the project**

**Make sure you have Nodejs installed on your machine as well.**

* **Create a folder for your project**
* **Open terminal in this folder**
* **>>npm init -y**
* **Install express using npm : >>npm i –save express**

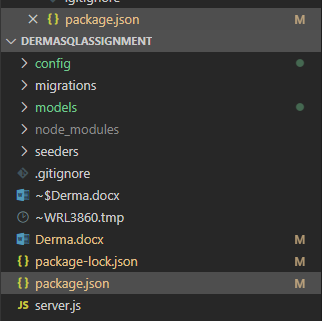
Install Sequelize- a library that allow us to connect via Node.js to our MySQL DB and operate in it, without writing SQL queries (handle it with JS syntax and operate in it as if we are handling objects).

In the root of the project

>> npm install –save sequelize

>> npm install --save sequelize-cli

>>npx sequelize init 🡺 will create folders and files in your project to hanle the SQL DB



Install .env :

dotenv - an npm package that loads environment variables from a .env file into [process.env](https://nodejs.org/docs/latest/api/process.html#process_process_env). Storing configuration in the environment separate from code is based on [The Twelve-Factor App](http://12factor.net/config) methodology.

Lets install it:

At root level-

> npm i dotenv

CREATE A NEW .env FILE AT ROOT LEVEL:

IN IT SET TWO ENV VARIABLES, AT THIS EXACT SYNTAX!

ROOT-> .env 🡺

\*\*make sure it’s in the .gitignore file as well!!!

Creating the server and app entry point:

**Create a new file in the root of the project 🡪 app.js**

const express  = require( 'express' );

const app = express();

app.use(express.json())

app.get('/', (req, res)=>{

    console.log('hi there,you got to / route');

    res.status(200).send('you got to / route');

});

module.exports = app

**Create a new file in the root of the project 🡪 server.js**

const http = require('http');

const dotenv = require('dotenv')

const app = require('./app');

dotenv.config();

const PORT = process.env.PORT || 3000;

const mode = process.env.NODE\_ENV;

const server = http.createServer(app);

server.listen(PORT, ()=>{

    console.log(`you are listenning to PORT: ${PORT}, in ${mode} environment`)

});

GIT- version control

Create a **.gitignore**  file

We will make sure you add =>

# dependencies

node\_modules

# misc.

.env

.env ->a file in which we will store some global variables that may contain sensitive information such as API keys we don’t want GitHub users to see.

**Initializing GitHub Repository:**

In the terminal :

We want to be on the root folder ->

>> git init

>> git add . 🡺 stage all files!

>> git status 🡺 check which files got staged

>> git commit -m 'Project first setup'

* Go to GitHub and open a new reposetory

>> git remote add origin “repo URL…”

>>git push -u origin master

**Routes & controllers**

Create folder at root level 🡪 controllers

Create userController.js file

Create folder at root level🡪 routes.

Create userRoutes.js file.

We will separate the request’s route and functionality into 2 separate files,

To fit mvc structure.

We will come back here later…

**for now lets install MySQL into our project using mysql2 npm package:**

[**https://www.npmjs.com/package/mysql2**](https://www.npmjs.com/package/mysql2)

**terminal at root level: >>npm i –save mysql2**

**now we will create a user model, it will include this data: userId, name, email, password.**

**This model will correspond with the user table which will be formatted with the same columns.**

**>>npx sequelize model:generate –name User –attributes name:string, email:string, password:string**

**This will create a user.js model file in model folder and a migration file which allows us to add more restriction for each column.**

**Id column is auto generated.**